EAST Search History

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
S1	83	donald near hooper.in.	US-PGPUB; USPAT	OR	ON	2006/12/12 16:00
S2	49	eric near walker.in.	US-PGPUB; USPAT	OR	ON	2006/12/12 16:01
S3	14713	intel.as.	US-PGPUB; USPAT	OR	ON	2006/12/12 16:01
S4	7	S3 and (instruction and operand and thread and register).clm.	US-PGPUB; USPAT	OR	ON	2006/12/12 16:06
S5	4	S3 and (instruction and operand and user).clm.	US-PGPUB; USPAT	OR	ON	2006/12/12 16:06
S6	2863	717/124-135.ccls.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/12 16:08
S8	210	S6 and (instruction and operand and debug$5)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/12 16:09
S9	96	(trac$4 near3 operand) with instruction	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/12 16:55
S10	28	S9 and debug$5	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/12 16:55
S11	21	S10 and (@pd<"20031113" or @ad<"20031113" or @prad<"20031113" or @rlad<"20031113")	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/12 16:55
S12	944	(operand or variable) with (second adj instruction)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/12 16:59

EAST Search History

S13	21	S12 and (trac$4 near3 operand)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/12 17:00
S14	3262	instruction adj (list or table or graph)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 07:46
S15	1615	processor near simulat$4	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 07:46
S16	46	S14 and S15	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 07:46
S17	3	"6611276".pn.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 07:59
S18	18	("3373408" | "3940745" | "5168554" | "5339415" | "5369570" | "5727209" | "5802371" | "5812133" | "5877764" | "6067641" | "6105051" | "6282701").PN. OR ("6611276").URPN.	US-PGPUB; USPAT; USOCR	OR	ON	2006/12/13 07:59
S19	4	("4730315" "4821220").pn.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 08:19

EAST Search History

S20	48	(US-20050071822-$ or US-20040088691-$ or US-20030192034-$ or US-20030110476-$ or US-20040030962-$ or US-20020144235-$ or US-20030088854-$ or US-20020004933-$ or US-20010034880-$ or US-20020199173-$ or US-20040162717-$).did. or (US-6792599-$ or US-7055136-$ or US-7028291-$ or US-6973417-$ or US-6804814-$ or US-6954923-$ or US-6826748-$ or US-6591378-$ or US-6557119-$ or US-6502210-$ or US-6463553-$ or US-6487683-$ or US-6434741-$ or US-6240544-$ or US-6009270-$ or US-5978584-$ or US-4819234-$ or US-4879646-$ or US-6006033-$ or US-5446876-$ or US-6088790-$ or US-5761474-$ or US-6487715-$ or US-6021261-$ or US-4951195-$ or US-7150002-$). did. or (US-7093249-$ or US-6971084-$ or US-6671827-$ or US-6282701-$ or US-6105051-$ or US-6067641-$ or US-5877764-$ or US-5727209-$ or US-5168554-$ or US-6611276-$ or US-4730315-$). did.	US-PGPUB; USPAT	OR	ON	2006/12/13 08:21
S21	4	S20 and (instruction near list$4)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 08:32
S23	6	S20 and (trac$4 near (operand or variable))	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 08:33
S24	18	S20 and (program$4 adj counter)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 08:36

EAST Search History

S25	7	S24 and tracing	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 08:36
S26	1	S20 and (operand adj dependenc$4)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 08:43
S28	118	tracing near (operand or variable or argument)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 10:15
S29	81	S28 and instruction	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 10:16
S30	2246	(operand or variable or argument) near dependenc$4	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 10:31
S32	41	S30 and (instruction near list)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 10:32
S33	22	("4594655"|"4734852"|"5128890"|"5488729"|"5838941"|"5839928"|"5887160"|"6145074"|"6256721"|"6430679"|"6857060").PN.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 10:41
S34	35212	(instruction or code) near (list$4 or chart$4 or graph$4 or table)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 11:41

EAST Search History

S35	116	S34 and ((operand or variable or argument or parameter) near dependenc$4)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 11:43
S36	81	S35 and user	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 11:43
S37	59	(US-20050071822-$ or US-20040088691-$ or US-20030192034-$ or US-20030110476-$ or US-20040030962-$ or US-20020144235-$ or US-20030088854-$ or US-20020004933-$ or US-20010034880-$ or US-20020199173-$ or US-20040162717-$ or US-20030154028-$ or US-20040255099-$).did. or (US-6792599-$ or US-7055136-$ or US-7028291-$ or US-6973417-$ or US-6804814-$ or US-6954923-$ or US-6826748-$ or US-6591378-$ or US-6557119-$ or US-6502210-$ or US-6463553-$ or US-6487683-$ or US-6434741-$ or US-6240544-$ or US-6009270-$ or US-5978584-$ or US-4819234-$ or US-4879646-$ or US-6006033-$ or US-5446876-$ or US-6088790-$ or US-5761474-$ or US-6487715-$ or US-6021261-$ or US-4951195-$ or US-7150002-$). did. or (US-7093249-$ or US-6971084-$ or US-6671827-$ or US-6282701-$ or US-6105051-$ or US-6067641-$ or US-5877764-$ or US-5727209-$ or US-5168554-$ or US-6611276-$ or US-4730315-$ or US-7093236-$ or US-7093108-$ or US-7080289-$ or US-6829733-$ or US-5881288-$ or US-5862336-$ or US-5857077-$ or US-7080365-$ or US-6145074-$).did.	US-PGPUB; USPAT	OR	ON	2006/12/13 11:49

EAST Search History

S38	9	S37 and (processor near simulat$4)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 11:49
S39	20	("20020122062" | "20020124042" | "20020124205" | "20020124241" | "5261097" | "5504881" | "5600789" | "5825361" | "5953530" | "6002868" | "6023773" | "6067639" | "6077304" | "6167455" | "6223228" | "6336088" | "6397378" | "6611276" | "6681384" | "6684385").PN.	US-PGPUB; USPAT; USOCR	OR	ON	2006/12/13 12:02
S40	3	("6687898" "6363523").pn.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 13:41
S41	1615	processor near simulat$4	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 14:51
S42	354	S41 and (instruction near (set or list$4))	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 14:51
S43	94	S42 and (user near3 (select$4 or choos$4 or pick$4))	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 14:52
S44	83	S43 and (operand or variable or argument or parameter)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 14:53
S45	2863	717/124-135.ccls.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 15:20

EAST Search History

S46	55	S45 and (processor near simulat$4)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 15:20
S47	869533	(profil$4 or (call adj structure))	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 15:37
S48	15588	S47 and (tracing or tracer)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 15:38
S49	8903	S48 and (operand or argument or parameter)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 15:39
S50	988	S49 and debug$5	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 15:40
S51	376	S50 and (instruction near (list$4 or set or group$4))	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 15:47
S52	6	S51 and (processor near simulat$4)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 15:44
S54	4029	(operand or parameter or arguement or variable) near dependenc$4	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 16:54

EAST Search History

S55	119	S54 and (tracing or tracer)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 16:41
S56	113	S55 and (@pd<"20031113" or @ad<"20031113" or @prad<"20031113" or @rlad<"20031113")	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 16:42
S57	4136	(operand or parameter or arguement or variable or pipeline) near dependenc$4	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 16:55
S58	7	S57 and (processor near simulat$4)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/13 16:55
S59	4	((program adj counter) or pc or pcv) with (operand near map$4)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 07:39
S60	244	operand near map$4	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 07:39
S61	209	S60 and register	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 07:40
S63	69	S61 and (program adj counter)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 07:40

EAST Search History

S64	39	("4430706" | "4755966" | "4858104" | "4891753" | "4914579" | "5072364" | "5093778" | "5129067" | "5136697" | "5193167" | "5201057" | "5222240" | "5283873" | "5283874" | "5287467" | "5295248" | "5327547" | "5333283").PN. OR ("5974538").URPN.	US-PGPUB; USPAT; USOCR	OR	ON	2006/12/14 07:51
S65	1685	(program adj counter) with cycle	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 08:44
S66	1065	(program adj counter) with instruction with cycle	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 08:44
S67	598	S66 and operand	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 08:45

EAST Search History

S68	75	(US-20030154028-$ or US-20040255099-$ or US-20020184613-$ or US-20010056341-$ or US-20030135718-$ or US-20050071822-$ or US-20040088691-$ or US-20040030962-$ or US-20030192034-$ or US-20030110476-$ or US-20030088854-$ or US-20020199173-$ or US-20020144235-$ or US-20020004933-$ or US-20010034880-$ or US-20040162717-$).did. or (US-4879646-$ or US-5446876-$ or US-5978584-$ or US-6006033-$ or US-6009270-$ or US-6088790-$ or US-6829733-$ or US-6701515-$ or US-7080289-$ or US-5881288-$ or US-5862336-$ or US-7093236-$ or US-6145074-$ or US-5857077-$ or US-5761474-$ or US-4819234-$ or US-7080365-$ or US-7093108-$ or US-7137105-$ or US-7134116-$ or US-7107578-$ or US-6848097-$ or US-6728949-$ or US-6493868-$ or US-6083281-$ or US-6871298-$). did. or (US-6487715-$ or US-6021261-$ or US-7150002-$ or US-7093249-$ or US-6971084-$ or US-6282701-$ or US-6105051-$ or US-6067641-$ or US-5877764-$ or US-5168554-$ or US-4730315-$ or US-7055136-$ or US-7028291-$ or US-6973417-$ or US-6954923-$ or US-6826748-$ or US-6804814-$ or US-6792599-$ or US-6240544-$ or US-6463553-$ or US-4951195-$ or US-6611276-$ or US-6502210-$ or US-6557119-$ or US-6487683-$ or US-6671827-$ or US-5727209-$). did. or (US-6434741-$ or US-6591378-$ or US-5852726-$ or US-5974538-$ or US-5333283-$ or US-6047351-$).did.	US-PGPUB; USPAT	OR	ON	2006/12/14 08:48
S69	29	S68 and (program adj counter)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 08:49

EAST Search History

S71	22	S69 and cycle	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 08:49
S72	2	S68 and ((program adj counter) with cycle)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 08:55
S74	4	S68 and ((program adj counter) same cycle)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 08:55
S75	567	(register adj type) same address	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 09:13
S76	35	S75 and ("i/o" near register)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 09:17
S77	4711	index adj register	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 09:18
S78	437	S77 and (register near type)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 09:19
S79	1	S78 and ("non-i/o" adj register)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 09:18

EAST Search History

S80	2459	"i/o" adj register	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 09:23
S81	307	S80 and (index adj register)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 09:24
S82	32	S81 and (register adj type)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 09:24
S84	88	determin$4 with (register adj type)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 09:56
S85	4	("i/o" and "non-i/o") adj register	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 14:17
S86	7	(instruction adj map$4) same (instruction adj type)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 10:48
S88	248	map$4 same (register near type)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 10:48
S89	39	S88 and (instruction near type)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 10:49

EAST Search History

S90	159	(user adj interface) and (tracing with variable)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 11:58
S91	2	S90 and (tracing adj option)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 11:53
S92	18	(user adj interface) and (tracing adj option)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 11:58
S94	5131	option with variable	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 12:17
S95	22	S90 and S94	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 12:17
S96	205	register adj history	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 13:59
S97	15	S96 and (index adj register)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 14:13
S99	158	(register adj type) with (detect$4 or determin$4)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 14:32

EAST Search History

S100	8	program adj counter adj history	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 16:43
S101	2869	717/124-135.ccls.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 16:14
S102	28	S101 and (index adj register)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 16:14
S103	8	(program or instruction) adj counter adj history	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/14 16:44
S105	99	("5964893" "5704034" "5737516" "5768575" "6223280" "6446094" "5781753" "6131157" "5623615" "5553256" "5655096" "5805849" "6101597" "6393550" "6848074" "20030028844" "5903719" "20050108689" "6463553" "5522053" "6070235" "6516409" "6055621" "5317720" "5333296" "5394529" "5481689" "4802086" "5228131" "5434986" "5450555" "5600811" "5675768" "5822788" "6055622" "6311260" "5675758" "5802339" "6195745" "5784552" "5896521" "6141791" "6499123" "4763245" "4991080" "5261107" "5355457" "5978910" "6223280" "6212629").pn.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/12/18 13:03

PORTAL USPTO

Subscribe (Full Service) Register (Limited Service, Free) Login

Search: ● The ACM Digital Library ○ The Guide

operand dependency debugging SEARCH

THE ACM DIGITAL LIBRARY

Feedback Report a problem Satisfaction survey

Terms used **operand** **dependency** **debugging** Found **5,321** of **193,448**

Sort results by	relevance	Save results to a Binder
Display results	expanded form	Search Tips
		Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 21 - 40 of 200 Result page: previous 1 **2** 3 4 5 6 7 8 9 10 next

Best 200 shown Relevance scale

21 Instantaneous current modeling in a complex VLIW processor core

Radu Muresan, Catherine Gebotys
May 2005 **ACM Transactions on Embedded Computing Systems (TECS)**, Volume 4 Issue 2
Publisher: ACM Press
Full text available: pdf(3.64 MB) Additional Information: full citation, abstract, references, index terms

Measuring and modeling instantaneous current consumption or current dynamics of a processor is important in embedded system designs, wireless communications, low-energy mobile computing, security of communications, and reliability. In this paper, we introduce a new instruction-level based macromodeling approach for instantaneous current consumption in a complex processor core along with new instantaneous current measurement techniques at the instruction and program level. Current consumption and ...

Keywords: Instruction-level current model, current and power measurement in a processor, instantaneous current model, power and energy model

22 Type-based verification of sssembly language for compiler debugging

Bor-Yuh Evan Chang, Adam Chlipala, George C. Necula, Robert R. Schneck
January 2005 **Proceedings of the 2005 ACM SIGPLAN international workshop on Types in languages design and implementation**
Publisher: ACM Press
Full text available: pdf(369.80 KB) Additional Information: full citation, abstract, references, citings, index terms

It is a common belief that *certifying compilation*, which typically verifies the well-typedness of compiler output, can be an effective mechanism for compiler debugging, in addition to ensuring basic safety properties. Bytecode verification is a fairly simple example of this approach and derives its simplicity in part by compiling to carefully crafted high-level bytecodes. In this paper, we seek to push this method to native assembly code, while maintaining much of the simplicity of byteco ...

Keywords: abstract interpretation, assembly code, bytecode verification, certified compilation, dependent types

23 Incremental dynamic semantics for language-based programming environments

G. E. Kaiser
April 1989 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 11 Issue 2
Publisher: ACM Press

Full text available: pdf(1.99 MB) Additional Information: full citation, abstract, references, citings, index terms, review

Attribute grammars are a formal notation for expressing the static semantics of programming languages—those properties that can be derived from inspection of the program text. Attribute grammars have become popular as a mechanism for generating language-based programming environments that incrementally perform symbol resolution, type checking, code generation, and derivation of other static semantic properties as the program is modified. However, attribute grammars are not suitable fo ...

24 Models and Measurements for Quality Assessment of Software
Siba N. Mohanty
September 1979 **ACM Computing Surveys (CSUR)**, Volume 11 Issue 3
Publisher: ACM Press
Full text available: pdf(1.95 MB) Additional Information: full citation, references, citings, index terms

25 Debugging of behavioral VHDL specifications by source level emulation
Gernot Koch, Udo Kebschull, Wolfgang Rosenstiel
December 1995 **Proceedings of the conference on European design automation**
Publisher: IEEE Computer Society Press
Full text available: pdf(630.04 KB) Additional Information: full citation, references, citings, index terms

26 Guidelines for creating a debuggable processor
R. E. McLear, D. M. Scheibelhut, E. Tammaru
March 1982 **ACM SIGARCH Computer Architecture News , ACM SIGPLAN Notices , Proceedings of the first international symposium on Architectural support for programming languages and operating systems ASPLOS-I**, Volume 10 , 17 Issue 2 , 4
Publisher: ACM Press
Full text available: pdf(687.40 KB) Additional Information: full citation, abstract, references, citings, index terms

Hardware without software is of little use. Systems that ease the task of debugging software reduce cost and speed development. This paper presents guidelines for designing processors that ease debugging for real-time computer systems. Special hardware can aid the debugging process by tracing execution and accesses to memory. Such hardware requires access to signals that may not be readily available. Other, less exotic hardware provides an interface to the programmer and other processors. T ...

27 Instruction issue logic for pipelined supercomputers
Shlomo Weiss, James E. Smith
January 1984 **ACM SIGARCH Computer Architecture News , Proceedings of the 11th annual international symposium on Computer architecture ISCA '84**, Volume 12 Issue 3
Publisher: ACM Press
Full text available: pdf(969.61 KB) Additional Information: full citation, abstract, references, citings, index terms

Basic principles and design tradeoffs for control of pipelined processors are first discussed. We concentrate on register-register architectures like the CRAY-1 where pipeline control logic is localized to one or two pipeline stages and is referred to as "instruction issue logic". Design tradeoffs are explored by giving designs for a variety of instruction issue methods that represent a range of complexity and sophistication. These vary from the original CRAY-1 issue logic to a ...

28 Debugging concurrent processes: a case study

J. M. Stone
June 1988 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1988 conference on Programming Language design and Implementation PLDI '88**, Volume 23 Issue 7
Publisher: ACM Press
Full text available: pdf(932.85 KB) Additional Information: full citation, abstract, references, citings, index terms

We present a case study that illustrates a method of debugging concurrent processes in a parallel programming environment. It uses a new approach called speculative replay to reconstruct the behavior of a program from the histories of its individual processes. Known time dependencies between events in different processes are used to divide the histories into dependence blocks. A graphical representation called a concurrency map displays possibilities for co ...

29 A time-stamping algorithm for efficient performance estimation of superscalar processors
Gabriel Loh
June 2001 **ACM SIGMETRICS Performance Evaluation Review , Proceedings of the 2001 ACM SIGMETRICS international conference on Measurement and modeling of computer systems SIGMETRICS '01**, Volume 29 Issue 1
Publisher: ACM Press
Full text available: pdf(1.11 MB) Additional Information: full citation, abstract, references, citings

The increasing complexity of modern superscalar microprocessors makes the evaluation of new designs and techniques much more difficult. Fast and accurate methods for simulating program execution on realistic and hypothetical processor models are of great interest to many computer architects and compiler writers. There are many existing techniques, from profile based runtime estimation to complete cycle-level simulations. Many researchers choose to sacrifice the speed of profiling for the accurac ...

30 Sifting out the mud: low level C++ code reuse
Bjorn De Sutter, Bruno De Bus, Koen De Bosschere
November 2002 **ACM SIGPLAN Notices , Proceedings of the 17th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications OOPSLA '02**, Volume 37 Issue 11
Publisher: ACM Press
Full text available: pdf(1.35 MB) Additional Information: full citation, abstract, references, citings, index terms

More and more computers are being incorporated in devices where the available amount of memory is limited. This contrasts with the increasing need for additional functionality and the need for rapid application development. While object-oriented programming languages, providing mechanisms such as inheritance and templates, allow fast development of complex applications, they have a detrimental effect on program size. This paper introduces new techniques to reuse the code of whole procedures at t ...

Keywords: code compaction, code size reduction

31 Multithreading and multiprocessing: Extensible control architectures
Greg Hoover, Forrest Brewer, Timothy Sherwood
October 2006 **Proceedings of the 2006 international conference on Compilers, architecture and synthesis for embedded systems CASES '06**
Publisher: ACM Press
Full text available: pdf(353.71 KB) Additional Information: full citation, abstract, references, index terms

Architectural advances of modern systems has often been at odds with control complexity, requiring significant effort in both design and verification. This is particularly true for sequential controllers, where machine complexity can quickly surpass designer ability. Traditional solutions to this problem require elaborate specifications that are difficult to maintain and extend. Further, the logic generated from these specifications bares no resemblance to the intended behavior and often fails t ...

Keywords: control architecture, specification methodology

32 A language implementation design for a multiprocessor computer system
P. Hibbard, A. Hisgen, T. Rodeheffer
April 1978 **Proceedings of the 5th annual symposium on Computer architecture**
Publisher: ACM Press
Full text available: pdf(669.15 KB) Additional Information: full citation, abstract, references, citings, index terms

Theoretical and experimental results have indicated that automatic decompositions can discover modest amounts of parallelism. These investigations have tended to ignore the practical problems of language run-time organization, such as synchronization, communication, memory organization, resource management, and input/output. This paper describes a language implementation effort which combines the investigation of implicit and explicit parallel decomposition facilities with the practical con ...

33 Secure program execution via dynamic information flow tracking
G. Edward Suh, Jae W. Lee, David Zhang, Srinivas Devadas
October 2004 **ACM SIGPLAN Notices , ACM SIGOPS Operating Systems Review , ACM SIGARCH Computer Architecture News , Proceedings of the 11th international conference on Architectural support for programming languages and operating systems ASPLOS-XI**, Volume 39 , 38 , 32 Issue 11 , 5 , 5
Publisher: ACM Press
Full text available: pdf(263.33 KB) Additional Information: full citation, abstract, references, citings, index terms

We present a simple architectural mechanism called dynamic information flow tracking that can significantly improve the security of computing systems with negligible performance overhead. Dynamic information flow tracking protects programs against malicious software attacks by identifying spurious information flows from untrusted I/O and restricting the usage of the spurious information.Every security attack to take control of a program needs to transfer the program's control to malevolent code. ...

Keywords: buffer overflow, format string, hardware tagging

34 GPGPU: general purpose computation on graphics hardware
David Luebke, Mark Harris, Jens Krüger, Tim Purcell, Naga Govindaraju, Ian Buck, Cliff Woolley, Aaron Lefohn
August 2004 **ACM SIGGRAPH 2004 Course Notes SIGGRAPH '04**
Publisher: ACM Press
Full text available: pdf(63.03 MB) Additional Information: full citation, abstract, citings

The graphics processor (GPU) on today's commodity video cards has evolved into an extremely powerful and flexible processor. The latest graphics architectures provide tremendous memory bandwidth and computational horsepower, with fully programmable vertex and pixel processing units that support vector operations up to full IEEE floating point precision. High level languages have emerged for graphics hardware, making this computational power accessible. Architecturally, GPUs are highly parallel s ...

35 Validating the intel pentium 4 microprocessor
Bob Bentley
June 2001 **Proceedings of the 38th conference on Design automation**
Publisher: ACM Press
Full text available: pdf(163.66 KB) Additional Information: full citation, abstract, references, citings, index terms

Developing a new leading edge IA-32 micro-processor is an immensely complicated undertaking. In the case of the Pentium© 4 processor, the microarchitecture is significantly more complex than any previous IA-32 microprocessor and the implementation borrows almost nothing from any previous implementation. This paper describes how we went about the task of finding bugs in the Pentium© 4 processor design prior to initial silicon, and what we found along the way.

36 A Fast Assembly Level Reverse Execution Method via Dynamic Slicing
Tankut Akgul, Vincent J. Mooney III, Santosh Pande
May 2004 **Proceedings of the 26th International Conference on Software Engineering ICSE '04**
Publisher: IEEE Computer Society
Full text available: pdf(227.56 KB) Additional Information: full citation, abstract, citings, index terms

One of the most time consuming parts of debugging istrying to locate a bug. In this context, there are two powerfuldebugging aids which shorten debug time considerably:reverse execution and dynamic slicing. Reverse executioneliminates the need for repetitive program restartsevery time a bug location is missed. Dynamic slicing, onthe other hand, isolates code parts that influence an erroneousvariable at a program point. In this paper, we presentan approach which provides assembly level reverse ex ...

37 Pipeline Architecture
C. V. Ramamoorthy, H. F. Li
March 1977 **ACM Computing Surveys (CSUR),** Volume 9 Issue 1
Publisher: ACM Press
Full text available: pdf(3.53 MB) Additional Information: full citation, references, citings, index terms

38 Experience in the design, implementation and use of PL-11, a programming language for the PDP-11
Robert D. Russell
March 1976 **ACM SIGPLAN Notices , Proceedings of the ACM SIGMINI/SIGPLAN interface meeting on Programming systems in the small processor environment SIGMINI '76,** Volume 11 Issue 4
Publisher: ACM Press
Full text available: pdf(787.25 KB) Additional Information: full citation, abstract, references, citings, index terms

PL-11 is a programming language for the PDP-11 family of computers designed and implemented as part of the OMEGA Project at CERN (the European Organization for Nuclear Research). Its purpose is to provide an effective tool for both physicists and systems programmers to use in building real-time data acquisition systems that are on-line to high-energy physics experiments. It is a fairly typical member of the PL-class of programming languages (44) which are based on the initial design of PL36 ...

39 The program dependence graph and its use in optimization
Jeanne Ferrante, Karl J. Ottenstein, Joe D. Warren
July 1987 **ACM Transactions on Programming Languages and Systems (TOPLAS),** Volume 9 Issue 3
Publisher: ACM Press
Full text available: pdf(2.51 MB) Additional Information: full citation, abstract, references, citings, index terms, review

In this paper we present an intermediate program representation, called the program dependence graph (PDG), that makes explicit both the data and control dependences for each operation in a program. Data dependences have been used to represent only the relevant data flow relationships of a program. Control dependences are introduced to analogously represent only the essential control flow relationships of a program. Control dependences are derived from the ...

40 Link-time binary rewriting techniques for program compaction

Bjorn De Sutter, Bruno De Bus, Koen De Bosschere
September 2005 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 27 Issue 5
Publisher: ACM Press
Full text available: pdf(1.37 MB) Additional Information: full citation, abstract, references, index terms

Small program size is an important requirement for embedded systems with limited amounts of memory. We describe how link-time compaction through binary rewriting can achieve code size reductions of up to 62&percent; for statically bound languages such as C, C&plus;&plus;, and Fortran, without compromising on performance. We demonstrate how the limited amount of information about a program at link time can be exploited to overcome overhead resulting from separate compilation. This is done with sc ...

Keywords: Program representation, binary rewriting, code abstraction, compaction, interprocedural analysis, linker, whole-program optimization

Results 21 - 40 of 200 Result page: previous 1 **2** 3 4 5 6 7 8 9 10 next